Introducción a la Bioinformática:

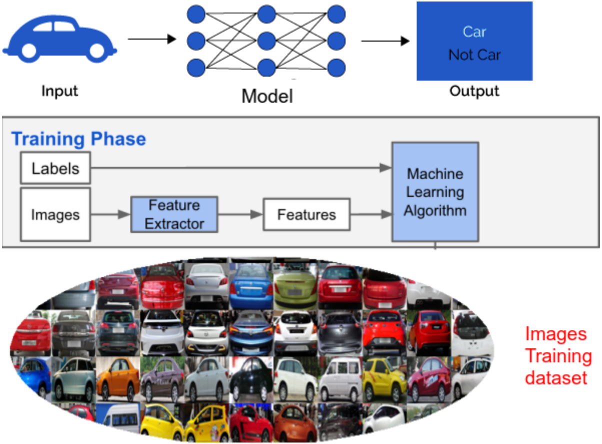
Data Redundancy and Clustering

Luis Garreta

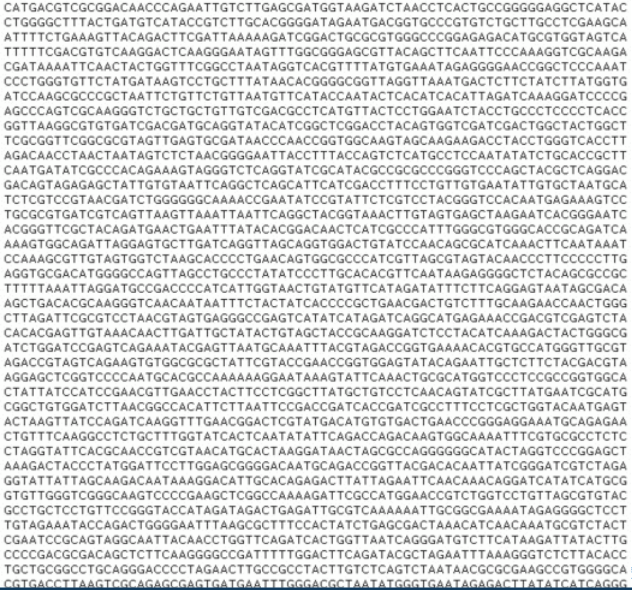
Doctorado en Ingeniería
Pontificia Universidad Javeriana – Cali

April 11, 2018

A Machine Learning Approach



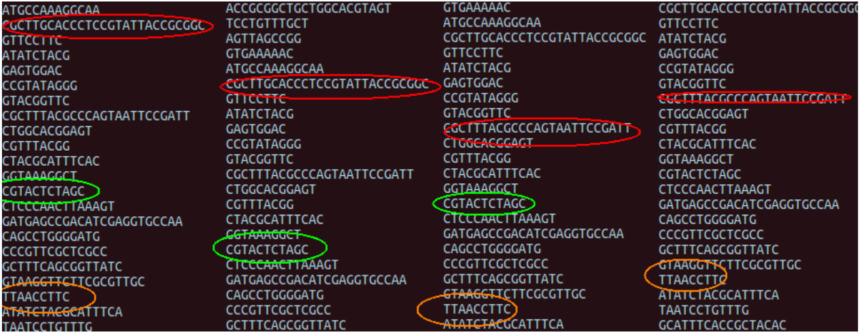
Sequences are the main data in Bioinformatics



```
CATGACGTCGCGGACAACCCAGAATTGTCTTGAGCGATGGTAAGATCTAACCTCACTGCCGGGGGAGGCTCATAC
CTGGGGCTTTACTGATGTCATACCGTCTTGCACGGGGATAGAATGACGGTGCCCGTGTCTGCTTGCCTCGAAGCA
ATTTTCTGAAAGTTACAGACTTCGATTAAAAAGATCGGACTGCGCGTGGGCCCGGAGAGACATGCGTGGTAGTCA
TTTTTCGACGTGTCAAGGACTCAAGGGAATAGTTTGGCGGGAGCGTTACAGCTTCAATTCCCAAAGGTCGCAAGA
CGATAAAATTCAACTACTGGTTTCGGCCTAATAGGTCACGTTTTATGTGAAATAGAGGGGAACCGGCTCCCAAAT
CCCTGGGTGTTCTATGATAAGTCCTGCTTTATAACACGGGGCGGTTAGGTTAAATGACTCTTCTATCTTATGGTG
ATCCAAGCGCCCGCTAATTCTGTTCTGTTAATGTTCATACCAATACTCACATCACATTAGATCAAAGGATCCCCG
AGCCCAGTCGCAAGGGTCTGCTGCTGTTGTCGACGCCTCATGTTACTCCTGGAATCTACCTGCCCTCCCCTCACC
GGTTAAGGCGTGTGATCGACGATGCAGGTATACATCGGCTCGGACCTACAGTGGTCGATCGACTGGCTACTGGCT
TCGCGGTTCGGCGCGTAGTTGAGTGCGATAACCCAACCGGTGGCAAGTAGCAAGAAGACCTACCTGGGTCACCTT
AGACAACCTAACTAATAGTCTCTAACGGGGAATTACCTTTACCAGTCTCATGCCTCCAATATATCTGCACCGCTT
CAATGATATCGCCCACAGAAAGTAGGGTCTCAGGTATCGCATACGCCGCGCCCGGGTCCCAGCTACGCTCAGGAC
GACAGTAGAGAGCTATTGTGTAATTCAGGCTCAGCATTCATCGACCTTTCCTGTTGTGAATATTGTGCTAATGCA
TCTCGTCCGTAACGATCTGGGGGGCAAAACCGAATATCCGTATTCTCGTCCTACGGGTCCACAATGAGAAAGTCC
TGCGCGTGATCGTCAGTTAAGTTAAATTAATTCAGGCTACGGTAAACTTGTAGTGAGCTAAGAATCACGGGAATC
ACGGGTTCGCTACAGATGAACTGAATTTATACACGGACAACTCATCGCCCATTTGGGCGTGGGCACCGCAGATCA
AAAGTGGCAGATTAGGAGTGCTTGATCAGGTTAGCAGGTGGACTGTATCCAACAGCGCATCAAACTTCAATAAAT
CCAAAGCGTTGTAGTGGTCTAAGCACCCCTGAACAGTGGCGCCCATCGTTAGCGTAGTACAACCCTTCCCCCTTG
AGGTGCGACATGGGGCCAGTTAGCCTGCCCTATATCCCTTGCACACGTTCAATAAGAGGGGCTCTACAGCGCCGC
TTTTTAAATTAGGATGCCGACCCCATCATTGGTAACTGTATGTTCATAGATATTTCTTCAGGAGTAATAGCGACA
AGCTGACACGCAAGGGTCAACAATAATTTCTACTATCACCCCGCTGAACGACTGTCTTTGCAAGAACCAACTGGG
CTTAGATTCGCGTCCTAACGTAGTGAGGGCCGAGTCATATCATAGATCAGGCATGAGAAACCGACGTCGAGTCTA
CACACGAGTTGTAAACAACTTGATTGCTATACTGTAGCTACCGCAAGGATCTCCTACATCAAAGACTACTGGGCG
ATCTGGATCCGAGTCAGAAATACGAGTTAATGCAAATTTACGTAGACCGGTGAAAACACGTGCCATGGGTTGCGT
AGACCGTAGTCAGAAGTGTGGCGCGCTATTCGTACCGAACCGGTGGAGTATACAGAATTGCTCTTCTACGACGTA
AGGAGCTCGGTCCCCAATGCACGCCAAAAAAGGAATAAAGTATTCAAACTGCGCATGGTCCCTCCGCCGGTGGCA
CTATTATCCATCCGAACGTTGAACCTACTTCCTCGGCTTATGCTGTCCTCAACAGTATCGCTTATGAATCGCATG
CGGCTGTGGATCTTAACGGCCACATTCTTAATTCCGACCGATCACCGATCGCCTTTCCTCGCTGGTACAATGAGT
ACTAAGTTATCCAGATCAAGGTTTGAACGGACTCGTATGACATGTGTGACTGAACCCGGGAGGAAATGCAGAGAA
CTGTTTCAAGGCCTCTGCTTTGGTATCACTCAATATATTCAGACCAGACAAGTGGCAAAATTTCGTGCGCCTCTC
CTAGGTATTCACGCAACCGTCGTAACATGCACTAAGGATAACTAGCGCCAGGGGGGCATACTAGGTCCCGGAGCT
AAAGACTACCCTATGGATTCCTTGGAGCGGGGACAATGCAGACCGGTTACGACACAATTATCGGGATCGTCTAGA
GGTATTATTAGCAAGACAATAAAGGACATTGCACAGAGACTTATTAGAATTCAACAAACAGGATCATATCATGCG
GTGTTGGGTCGGGCAAGTCCCCGAAGCTCGGCCAAAAGATTCGCCATGGAACCGTCTGGTCCTGTTAGCGTGTAC
GCCTGCTCCTGTTCCGGGTACCATAGATAGACTGAGATTGCGTCAAAAAATTGCGGCGAAAATAGAGGGGCTCCT
TGTAGAAATACCAGACTGGGGAATTTAAGCGCTTTCCACTATCTGAGCGACTAAACATCAACAAATGCGTCTACT
CGAATCCGCAGTAGGCAATTACAACCTGGTTCAGATCACTGGTTAATCAGGGATGTCTTCATAAGATTATACTTG
CCCCGACGCGACAGCTCTTCAAGGGGCCGATTTTTGGACTTCAGATACGCTAGAATTTAAAGGGTCTCTTACACC
TGCTGCGGCCTGCAGGGACCCCTAGAACTTGCCGCCTACTTGTCTCAGTCTAATAACGCGCGAAGCCGTGGGGCA
CGTGACCTTAAGTCGCAGAGCGAGTGATGAATTTGGGACGCTAATATGGGTGAATAGAGACTTATATCATCAGGG
```

But, they can be very redundant

Biolgical Sequences: DNA, ARN, Proteins, Motifs, Genes, Genomes, ...



```
ATGCCAAAGGCAA
CGCTTGCACCCTCCGTATTACCGCGGC
GTTCCTTC
ATATCTACG
GAGTGGAC
CCGTATAGGG
GTACGGTTC
CGCTTTACGCCCAGTAATTCCGATT
CTGGCACGGAGT
CGTTTACGG
CTACGCATTTCAC
GGTAAAGGCT
CGTACTCTAGC
CTCCCAACTTAAAGT
GATGAGCCGACATCGAGGTGCCAA
CAGCCTGGGGATG
CCCGTTCGCTCGCC
GCTTTCAGCGGTTATC
GTAAGGTTCTTCGCGTTGC
TTAACCTTC
ATATCTACGCATTTCA
TAATCCTGTTTG
```

```
ACCGCGGCTGCTGGCACGTAGT
TCCTGTTTGCT
AGTTAGCCGG
GTGAAAAAC
ATGCCAAAGGCAA
CGCTTGCACCCTCCGTATTACCGCGGC
GTTCCTTC
ATATCTACG
GAGTGGAC
CCGTATAGGG
GTACGGTTC
CGCTTTACGCCCAGTAATTCCGATT
CTGGCACGGAGT
CGTTTACGG
CTACGCATTTCAC
GGTAAAGGCT
CGTACTCTAGC
CTCCCAACTTAAAGT
GATGAGCCGACATCGAGGTGCCAA
CAGCCTGGGGATG
CCCGTTCGCTCGCC
GCTTTCAGCGGTTATC
```

```
GTGAAAAAC
ATGCCAAAGGCAA
CGCTTGCACCCTCCGTATTACCGCGGC
GTTCCTTC
ATATCTACG
GAGTGGAC
CCGTATAGGG
GTACGGTTC
CGCTTTACGCCCAGTAATTCCGATT
CTGGCACGGAGT
CGTTTACGG
CTACGCATTTCAC
GGTAAAGGCT
CGTACTCTAGC
CTCCCAACTTAAAGT
GATGAGCCGACATCGAGGTGCCAA
CAGCCTGGGGATG
CCCGTTCGCTCGCC
GCTTTCAGCGGTTATC
GTAAGGTTCTTCGCGTTGC
TTAACCTTC
ATATCTACGCATTTCA
```

```
CGCTTGCACCCTCCGTATTACCGCGGC
GTTCCTTC
ATATCTACG
GAGTGGAC
CCGTATAGGG
GTACGGTTC
CGCTTTACGCCCAGTAATTCCGATT
CTGGCACGGAGT
CGTTTACGG
CTACGCATTTCAC
GGTAAAGGCT
CGTACTCTAGC
CTCCCAACTTAAAGT
GATGAGCCGACATCGAGGTGCCAA
CAGCCTGGGGATG
CCCGTTCGCTCGCC
GCTTTCAGCGGTTATC
GTAAGGTTCTTCGCGTTGC
TTAACCTTC
ATATCTACGCATTTCA
TAATCCTGTTTG
GCATTTCACCGCTACAC
```

What kind of problems may this originate?

Problems with Data Redundancy

- Large number of inputs for computer algorithms:
  - ✓ Lower performance of computer algorithms
  - ✓ Computer algorithms will take more time (hours, days, months,...)
- Too many similar/repeated data samples:
  - ✓ Algorithms wont work in an optimal manner
  - ✓ Biased analysis
  - ✓ Lower ability to generalize (machine learning)

Reducing data set is therefore a very important step in Bioinformatics

Benefits of Reducing Data

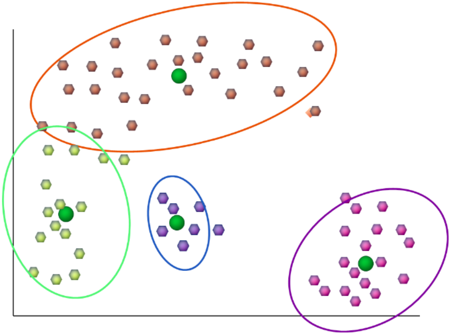
- Reducing the size of the data can result in:
  - ✓ Avoid noisy and redundant data
  - ✓ Increasing capabilities and generalization properties
  - ✓ Reducing space complexity of the classification problem
  - ✓ Decreasing the required computational time
- How to reduce data?

Two approaches for Reducing Biological Data

Data reduction can be achieved by Reducing instances and by Reducing features.

Reducing Instances

- Becomes especially important in case of **large data sets**.
- Storage and complexity constraints become **computationally expensive**.

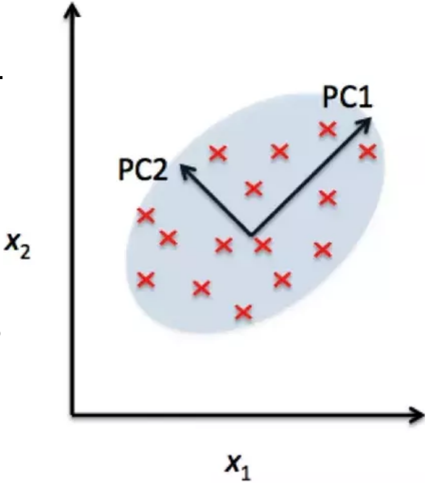


Reducing Features

Remove features that are **irrelevant** for the problem:

- Non meaningful for the final results
- Introduces **noise** to the analysis
- Resulting in **negative influence** for the analysis
- Add computational complexity to the algorithms.

What algorithms ?



...Algorithms

Algorithms for Reducing Data

- Algorithms for Reducing Instances
  - ✓ Agglomerative: Hierarchical (Clustering)
  - ✓ Partitioning: K-Means (Clustering)
  - ✓ Greedy: CD-Hit (Selecting) Hobohm and Sander's Algorithm
- Algorithms for Reducing Features:
  - ✓ Principal Component Analysis (PCA) for reducing dimensionality

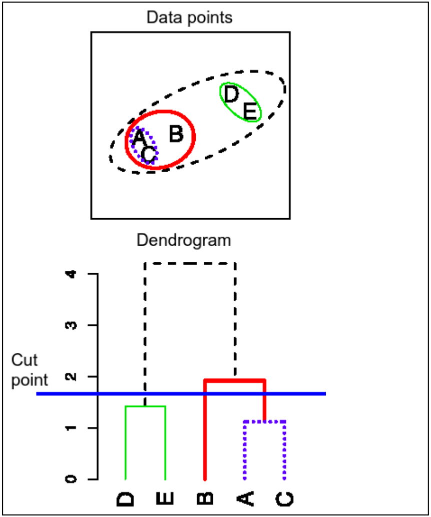
...Hierarchical Clustering

Clustering Data

Hierarchical Clustering (HC)

Given a set of data points:

- HC produces a sequence of clusterings that can be represented as a dendrogram.
- Each interior node of the dendrogram correspond to a mergin of two clusters (or points)
- A cut point is defined to get the number of clusters



Hierarchical Clustering Algorithm

Steps:

Let X = $x_1, \ldots, x_n$ be the set of data points.

1. Start with each item $x_1, \ldots, x_n$ in its own cluster $c_1, \ldots, c_n$
2. Let the distances (similarities) between the clusters equal the distances (similarities) between the items they contain.
3. Repeat:
   - 3.1 Merge the closest pair of clusters (c_i, c_j) into a single cluster, so that now you have one less cluster.
   - 3.2 Compute distance (similarities) between new cluster and each of the old clusters

Computing Distances in Hierarchical Clustering

- Computing distances can be done in different ways:
  - ✓ Single-linkage
  - ✓ Complete-linkage
  - ✓ Average-linkage (= UPGMA)
  - ✓ Others (from R hclust):
    - “"ward.D"”
    - “"ward.D2"”,
    - “"mcquitty"” (= WPGMA),
    - “"median"” (= WPGMC) or “"centroid"” (= UPGMC).

Common Distance Linkages

Single Linkage

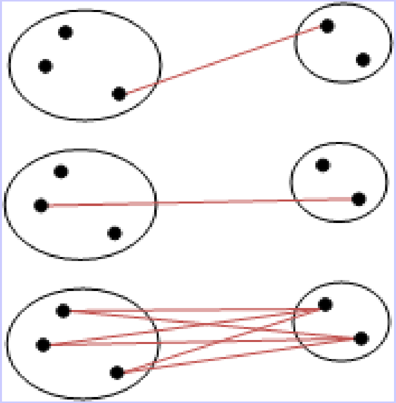
$$d(x,x')=\min_{x\in C_i,x'\in C_j} d(x,x')$$

Complete Linkage

$$d(x,x')=\max_{x\in C_i,x'\in C_j} d(x,x')$$

Average Linkage

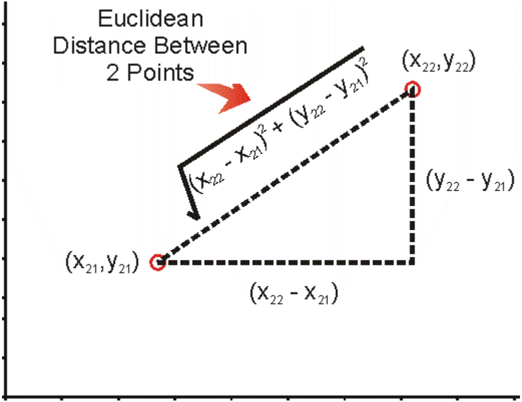
$$d(x,x')=\frac{\sum x\in C_i,x'\in C_j\, d(x,x')}{|C_i|.|C_j|}$$



Metrics to Calculate Distances: Euclidean Distance

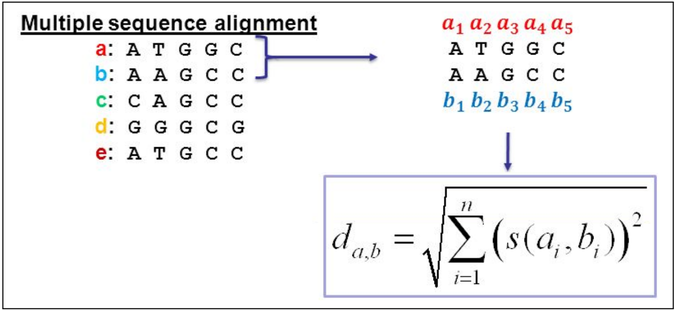
There are many metrics to calculate a distance between 2 points: p(x1, y1) and q(x2, y2) in xy-plane.

- Euclidean
- Manhattan
- Chebyshev


Euclidean Distance Between 2 Points
$\sqrt{(x_{22} - x_{21})^2 + (y_{22} - y_{21})^2}$
(x_{22}, y_{22})
$(y_{22} - y_{21})$
(x_{21}, y_{21})
$(x_{22} - x_{21})$

Metrics to Calculate Distances: Pairwise Distance

- The distance betwee each pair of sequences is based on Multiple Sequence Alignment



Multiple sequence alignment

```
a: A T G G C
b: A A G C C
c: C A G C C
d: G G G C G
e: A T G C C
```

$a_1\ a_2\ a_3\ a_4\ a_5$

```
A T G G C
A A G C C
```

$b_1\ b_2\ b_3\ b_4\ b_5$

$$d_{a,b} = \sqrt{\sum_{i=1}^{n} \left(s(a_i, b_i)\right)^2}$$

- $s(a_i, b_i)$ is the distance between **two nucleotides** (differences) or the distance between **two amino acids** (using PAM or BLOSSUM matrices).

Example Hierarchical Clustering

Clustering of distances in miles between U.S. cities

Input distance Matrix:

	BOS	NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS	0	206	429	1504	963	2976	3095	2979	1949
NY	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
MIA	1504	1308	1075	0	1329	3273	3053	2687	2037
CHI	963	802	671	1329	0	2013	2142	2054	996
SEA	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
DEN	1949	1771	1616	2037	996	1307	1235	1059	0

Example Hierarchical Clustering

After merging BOS with NY:

	BOS/NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY	0	223	1308	802	2815	2934	2786	1771
DC	223	0	1075	671	2684	2799	2631	1616
MIA	1308	1075	0	1329	3273	3053	2687	2037
CHI	802	671	1329	0	2013	2142	2054	996
SEA	2815	2684	3273	2013	0	808	1131	1307
SF	2934	2799	3053	2142	808	0	379	1235
LA	2786	2631	2687	2054	1131	379	0	1059
DEN	1771	1616	2037	996	1307	1235	1059	0

Example Hierarchical Clustering

After merging DC with BOS-NY:

	BOS/NY/DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY/DC	0	1075	671	2684	2799	2631	1616
MIA	1075	0	1329	3273	3053	2687	2037
CHI	671	1329	0	2013	2142	2054	996
SEA	2684	3273	2013	0	808	1131	1307
SF	2799	3053	2142	808	0	**379**	1235
LA	2631	2687	2054	1131	**379**	0	1059
DEN	1616	2037	996	1307	1235	1059	0

Example Hierarchical Clustering

After merging SF with LA:

	BOS/NY/DC/	MIA	CHI	SEA	SF/LA	DEN
BOS/NY/DC	0	1075	671	2684	2631	1616
MIA	1075	0	1329	3273	2687	2037
CHI	671	1329	0	2013	2054	996
SEA	2684	3273	2013	0	808	1307
SF/LA	2631	2687	2054	808	0	1059
DEN	1616	2037	996	1307	1059	0

Example Hierarchical Clustering

After merging CHI with BOS/NY/DC:

	BOS/NY/DC/CHI	MIA	SEA	SF/LA	DEN
BOS/NY/DC/CHI	0	1075	2013	2054	996
MIA	1075	0	3273	2687	2037
SEA	2013	3273	0	**808**	1307
SF/LA	2054	2687	**808**	0	1059
DEN	996	2037	1307	1059	0

Example Hierarchical Clustering

After merging SEA with SF/LA:

	BOS/NY/DC/CHI	MIA	SF/LA/SEA	DEN
BOS/NY/DC/CHI	0	1075	2013	996
MIA	1075	0	2687	2037
SF/LA/SEA	2054	2687	0	1059
DEN	996	2037	1059	0

Example Hierarchical Clustering

After merging DEN with BOS/NY/DC/CHI:

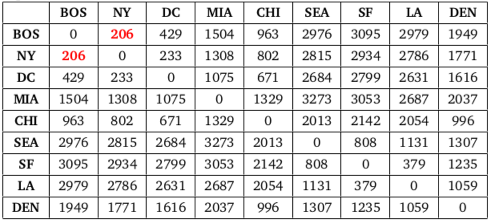
	BOS/NY/DC/CHI/DEN	**MIA**	**SF/LA/SEA**
BOS/NY/DC/CHI/DEN	0	1075	1059
MIA	1075	0	2687
SF/LA/SEA	1059	**2687**	0

Example Hierarchical Clustering

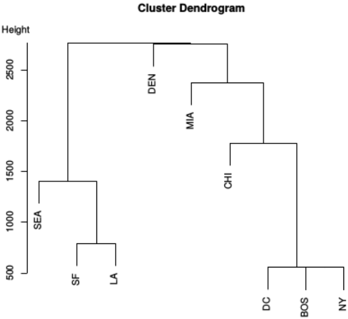
After merging SF/LA/SEA with BOS/NY/DC/CHI/DEN:

	BOS/NY/DC/CHI/ DEN/SF/LA/SEA	MIA
BOS/NY/DC/CHI/ DEN/SF/LA/SEA	0	1075
MIA	1075	0

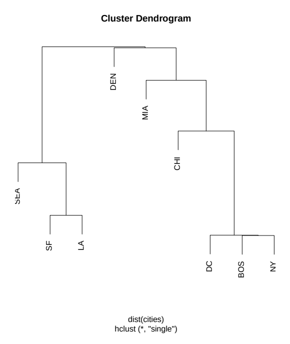
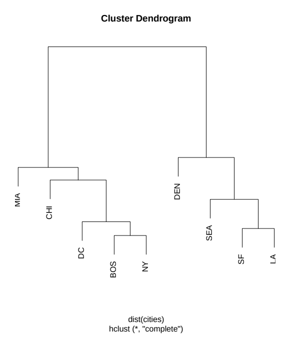
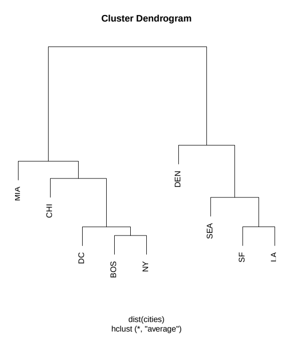
Dendrogram



	BOS	NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS	0	206	429	1504	963	2976	3095	2979	1949
NY	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
MIA	1504	1308	1075	0	1329	3273	3053	2687	2037
CHI	963	802	671	1329	0	2013	2142	2054	996
SEA	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
DEN	1949	1771	1616	2037	996	1307	1235	1059	0



Dendograms using different distance methods

Single	Complete	Average
 	 	

Hierarchical Clustering Considerations

Advantages

1. No apriori information about the number of clusters required.
2. Easy to implement and gives best result in some cases.

Hierarchical Clustering Considerations

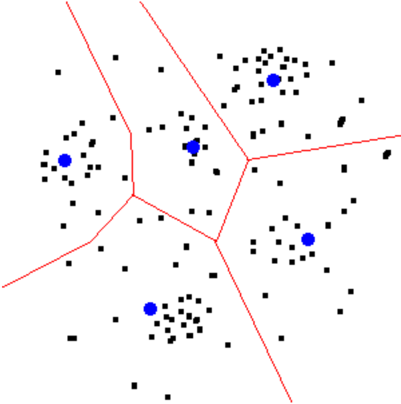
Disadvantages

1. Algorithm can never undo what was done previously.
2. Time complexity of at least $O(n^2 logn)$ is required, where ‘n’ is the number of data points.
3. Algorithms can suffer with one or more of the following:
   - 3.1 Sensitivity to noise and outliers
   - 3.2 Breaking large clusters
4. No objective function is directly minimized

Sometimes it is difficult to identify the correct number of clusters by the dendogram.

K-means Properties

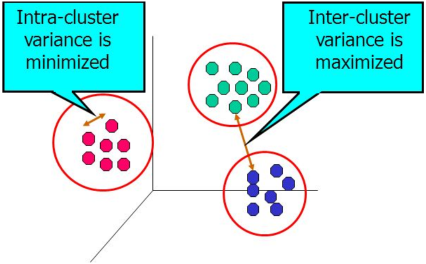
- One of the most important partitioning Method
- Different concept than the hierarchical clustering
- Not based on distance measures (e.g. Euclidean)
- Instead a direct measure distance, it uses the **whitin-cluster variation**



K-means within-cluster variation

- Without-cluster variation:
  - ✓ It is basically the (squared) distance from each observation to the center of the associated cluster.
  - ✓ Try to form homogeneous clusters
  - ✓ Segmenting the data
- Then, selection of distance measure is not needed
  - ✓ Minimize within cluster variation
  - ✓ Maximize between cluster variation

s

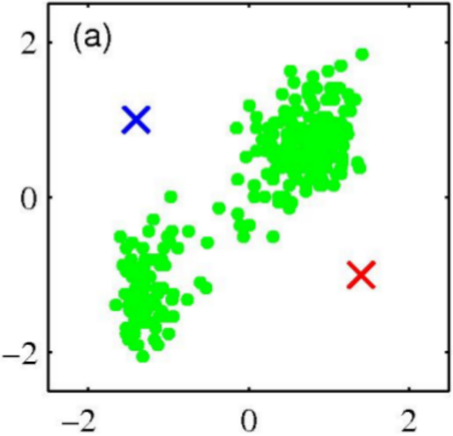

Intra-cluster variance is minimized
Inter-cluster variance is maximized

K-means Algorithm (Lloyd's Algorithm)

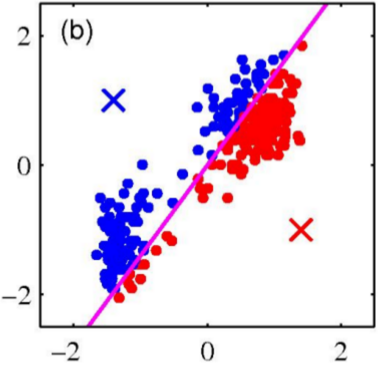
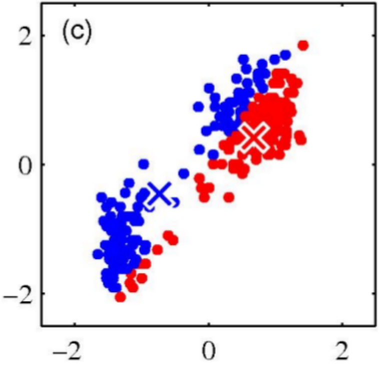
1. The clustering process starts by randomly assigning ***K*** objects to a number of clusters.
2. Repeat until cluster centers keep changing
   2.1 Compute the distance from each data point to the current cluster center $C_i, 1 \leq i \leq K$ and assign the point to the nearest cluster
   2.2 After the assignment of all data points, compute new centers for each cluster by taking the centroid of all the points in that cluster
   2.3 If the reallocation of an object to another cluster decreases the within-cluster variation, this object is reassigned to that cluster.
3. Output cluster centers and assignments

K-means Example:
Define initial K clusters

Pick ***K*** random points as cluster centers (means)

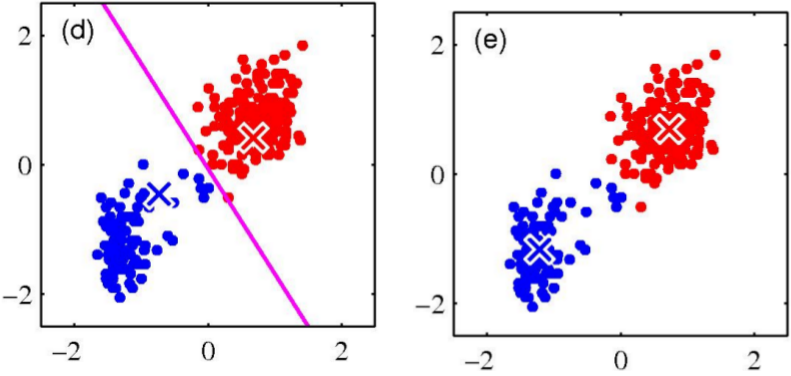


K-means Example: *Iterative Step1 and Step2*

Step1	Step2
Assign data points to closest cluster center	Change the cluster center to the average of the assigned points
	

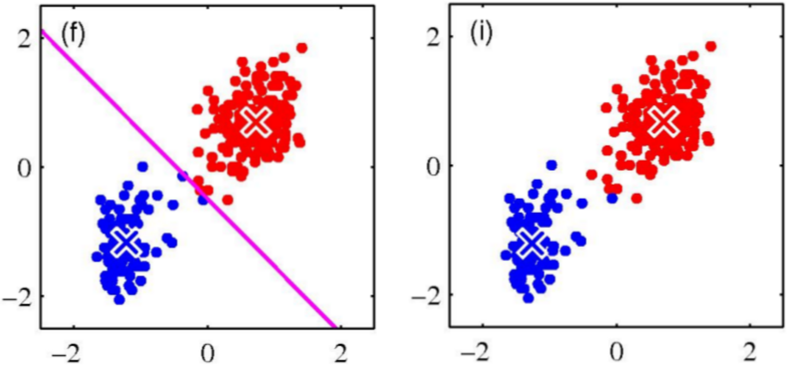
K-means Example:

Repeat until convergence



K-means Example:

Repeat until convergence

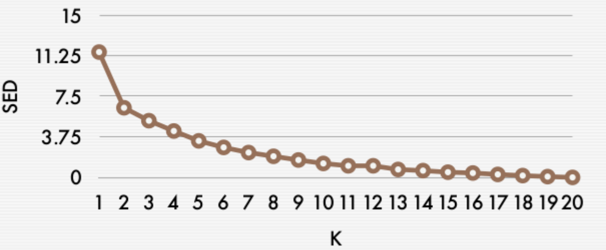


K-means Algorithm: *Parameters*

- K-means partitions a set of N points into K clusters
- Each cluster is represented with a mean (a centroid o "k-means")
- **Input**:
  - ✓ A set **V** with **N** points $(v_1, v_2, ..., v_n)$, and
  - ✓ The desired number of clusters ***K,*** and
  - ✓ A distance measure between any two points $d(v, w)$
- **Output:**
  - ✓ A set ***X*** of ***K*** cluster centers that minimize the squared error distortion
  - ✓ The squared error distortion of Data and Centers, is defined as the mean squared distance from each data point to its nearest center,

How To Choose K?

- The simplest approach is to start with K=1 and increase K until the squared error distortion (SED) stops decreasing

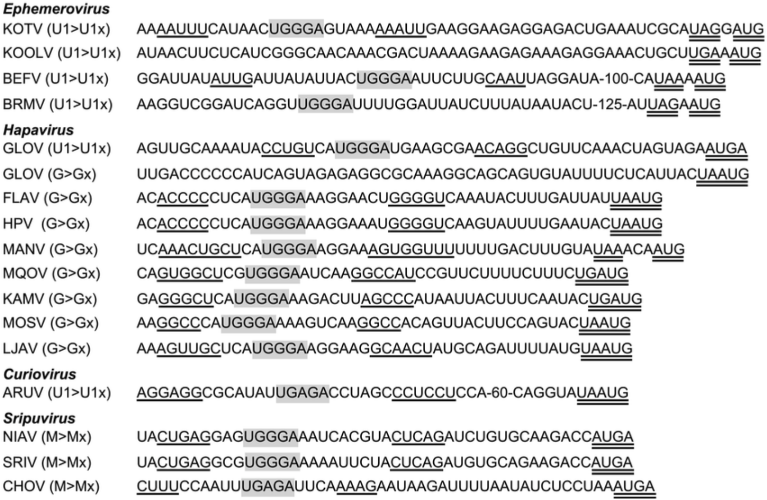


Representative Datasets

Drawbacks of Clustering Approaches for Large Datasets

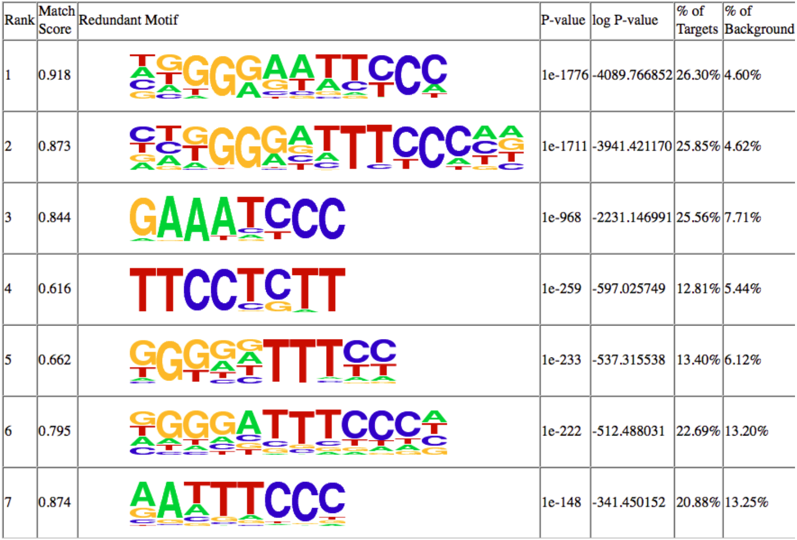
- In Bioinformatics:
  - ✓ Size of datasets commonly seen in current research (millions to billions of sequences)
- Previous clustering approaches:
  - ✓ Require the computation of all pairwise distances between a set of sequences,
  - ✓ i.e. the running time is proportional to n^2 , where n is the number of sequences.
- Given the large size of bioinformatics datasets:
  - ✓ Such algorithms are impractical!

Sequence motifs in the genomes of sripuviruses, curioviruses, hapaviruses and ephemeroviruse



Ephemerovirus	
KOTV (U1>U1x)	AAAAUUUCAUAACUGGGAGUAAAAAAUUGAAGGAAGAGGAGACUGAAAUCGCAUAGGAUG
KOOLV (U1>U1x)	AUAACUUCUCAUCGGGCAACAAACGACUAAAAGAAGAGAAAGAGGAAACUGCUUGAAAUG
BEFV (U1>U1x)	GGAUUAUAUUGAUUAUAUUACUGGGAAUUCUUGCAAUUAGGAUA-100-CAUAAAAUG
BRMV (U1>U1x)	AAGGUCGGAUCAGGUUGGGAUUUUGGAUUAUCUUUAUAAUACU-125-AUUAGAAUG
Hapavirus	
GLOV (U1>U1x)	AGUUGCAAAAUACCUGUCAUGGGAUGAAGCGAACAGGCUGUUCAAACUAGUAGAAUGA
GLOV (G>Gx)	UUGACCCCCCAUCAGUAGAGAGGCGCAAAGGCAGCAGUGUAUUUUCUCAUUACUAAUG
FLAV (G>Gx)	ACACCCCCUCAUGGGAAAGGAACUGGGGUCAAAUACUUUGAUUAUUAAUG
HPV (G>Gx)	ACACCCCCUCAUGGGAAAGGAAAUGGGGUCAAGUAUUUUGAAUACUAAUG
MANV (G>Gx)	UCAAACUGCUCAUGGGAAGGAAAGUGGUUUUUUUGACUUUGUAUAAACAAUG
MQOV (G>Gx)	CAGUGGCUCGUGGGAAUCAAGGCCAUCCGUUCUUUUCUUUCUGAUG
KAMV (G>Gx)	GAGGGCUCAUGGGAAAGACUUAGCCCAUAAUUACUUUCAAUACUGAUG
MOSV (G>Gx)	AAGGCCCAUGGGAAAAGUCAAGGCCACAGUUACUUCCAGUACUAAUG
LJAV (G>Gx)	AAAGUUGCUCAUGGGAAGGAAGGCAACUAUGCAGAUUUUAUGUAAUG
Curiovirus	
ARUV (U1>U1x)	AGGAGGCGCAUAUUGAGACCUAGCCCUCCUCCA-60-CAGGUAUAAUG
Sripuvirus	
NIAV (M>Mx)	UACUGAGGAGUGGGAAAUCACGUACUCAGAUCUGUGCAAGACCAUGA
SRIV (M>Mx)	UACUGAGGCGUGGGAAAAAUUCUACUCAGAUGUGCAGAAGACCAUGA
CHOV (M>Mx)	CUUUCCAAUUUGAGAUUCAAAAGAAUAAGAUUUUAAUAUCUCCUAAAUGA

True Motifs or Noisy Sequences?



Rank	Match Score	Redundant Motif	P-value	log P-value	% of Targets	% of Background
1	0.918		1e-1776	-4089.766852	26.30%	4.60%
2	0.873		1e-1711	-3941.421170	25.85%	4.62%
3	0.844		1e-968	-2231.146991	25.56%	7.71%
4	0.616		1e-259	-597.025749	12.81%	5.44%
5	0.662		1e-233	-537.315538	13.40%	6.12%
6	0.795		1e-222	-512.488031	22.69%	13.20%
7	0.874		1e-148	-341.450152	20.88%	13.25%

Representative Data Sets

- In sequence analysis **a number of algorithms** exist for selecting a representative subset from a set of data points.
- This is generally done by **keeping only one of two very similar data points**.
- In order to do this a measure for **similarity must be defined** between two data points:
  - ✓ e.g., percentage identity, alignment score, or significance of alignment score.
- Hobohm et al. [1992] have presented two algorithms for making a representative set from a list of data points D.

Hobohm1 Algorithm

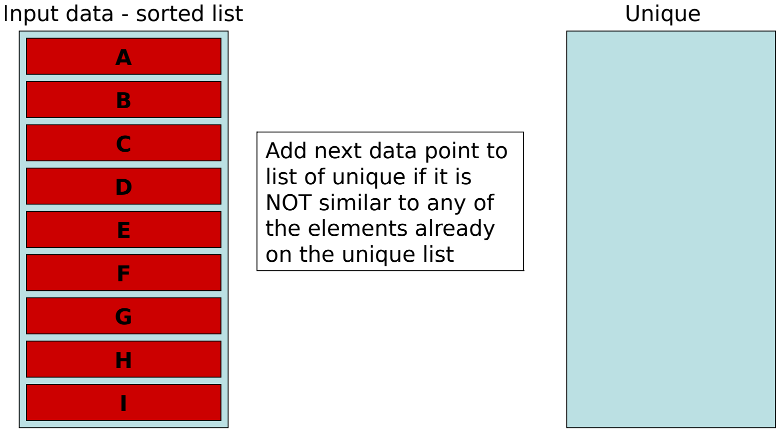
- Fast
- Requires a prior sorting of data

Algorithm:

Repeat for all data points on the list D:

- Add next data point in D to list of non-redundant data points N if it is not similar to any of the elements already on the list.

Hobohm1: Start with an empty and sorted list



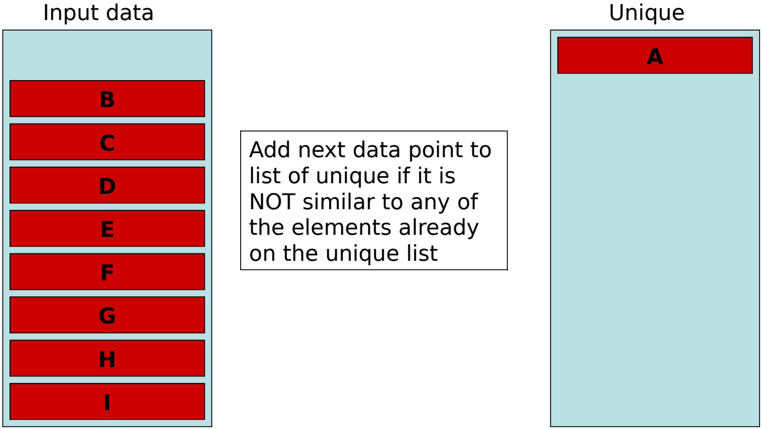
Input data - sorted list

A
B
C
D
E
F
G
H
I

Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

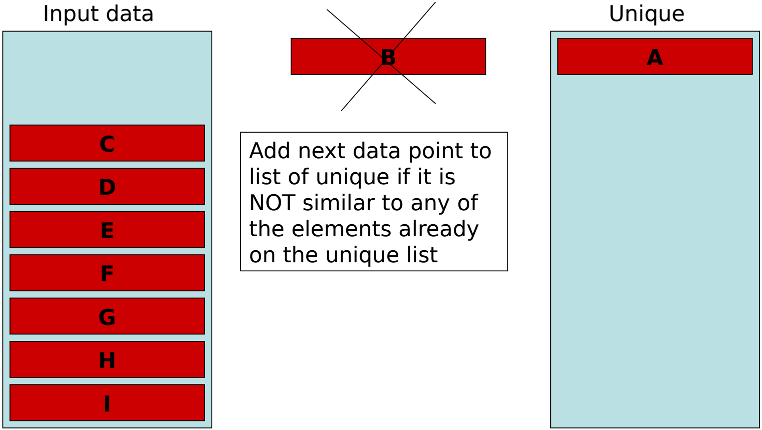
Unique

Hobohm1: Add First Element Without Any Comparison



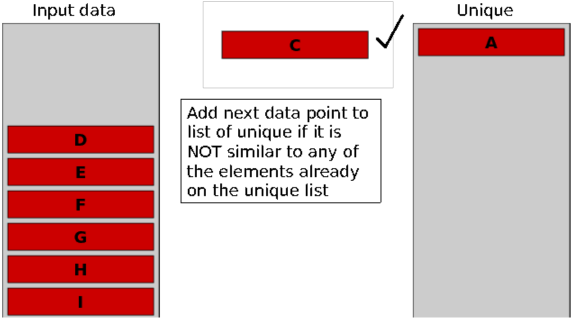
Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

Hobohm1: Add Next One Comparing with the First One



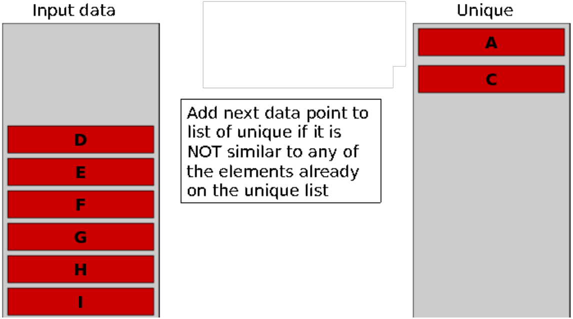
Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

Hobohm1: Compare the next one with the current representatives



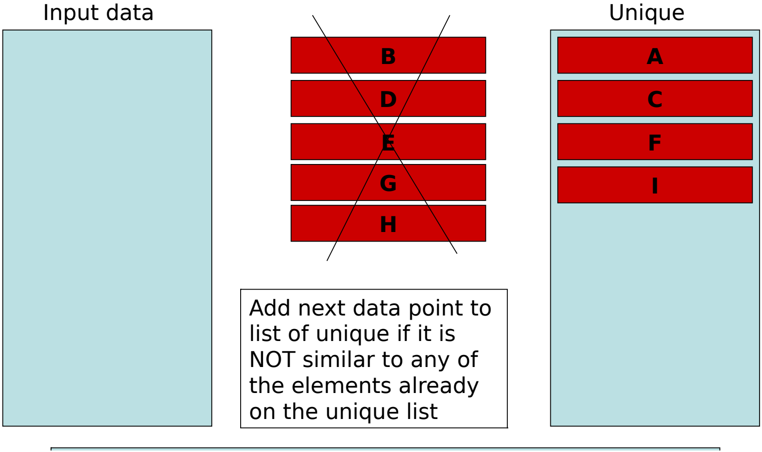
Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

Hobohm1: Create a new cluster if not similar



Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

Hobohm1: Add the Other Ones and Compare



Add next data point to list of unique if it is NOT similar to any of the elements already on the unique list

Hobohm1: Considerations

- Before applying the algorithm, **the data points can be sorted** according to some **property**.
- So, **maximizing the average value of this property** in the selected set:
  - ✓ Points higher on the list have **less chance of being filtered out**.
- The property can, e.g., the **length** of the sequence

CD-Hit: Fast Clustering for Large Datasets (Databases)

Greedy Clustering

- An alternative is provided by greedy clustering algorithms, exemplified by CD-hit (Li et al.)

CD-Hit Overview

- CDHIT is a program commonly **used to cluster nucleotide/protein sequences**.
- It is **used routinely by NCBI** to get rid of redundant sequences in the NR (non-redundant) database.
- **It is extremely fast** compared to a traditional all vs all blast and subsequent pair-wise clustering.
- **CDHIT doesn't use dynamic programming** to determine sequence similarity.
  - ✓ That's probably the biggest reason for it's speed.
  - ✓ It looks strictly at exact sequence **identity of k-mers**.

CD-Hit: *The basic algorithm*

1. Sort the sequences in decreasing order of their lengths
2. Pick the first sequence not assigned to a cluster :
   - ✓ This sequence becomes the center of a new cluster
3. Compare all unasigned sequences to already computed cluster centers:
   - ✓ First using a quick k-mer distance approach,
   - ✓ Then checking the promising alignments with a full smith-waterman algorithm.
   - ✓ If any of the sequence falls above the threshold for similarity, then it is grouped together with the first sequence.
4. Repeat from 2.

The output cluster sequences are the longest sequence out of each cluster group.

Exercise

Implements de CD-Hit Algorithm
Resources:

- CD-Hit is explained at:
  `http://blog.nextgenetics.net/?e=26`
- Some sequences for testing at:
  `http://github.com/lgarreta/bioinformatica/04-clustering/`